```
SELECT PERSONNEL-FILE
    ASSIGN                 TO INFILE
    ORGANIZATION           IS INDEXED
    ACCESS MODE            IS RANDOM
    RECORD KEY             IS EMPLOYEE-NUMBER
    ALTERNATE RECORD KEY IS EMPLOYEE-AGE WITH DUPLICATES
    ALTERNATE RECORD KEY IS DEPARTMENT-CODE WITH DUPLICATES
    FILE STATUS            IS PERSONNEL-FILE-STATUS.
```

READ Statement. An indexed file can be read in sequential or random access modes. Sequential input requires the exact same syntax of the READ statement for sequential files, with the exception of the NEXT clause. The syntax of this format is as follows.

/

```
READ file-name [ NEXT ] RECORD [ INTO identifier ]
    [ AT END imperative-statement-1 ]
    [ NOT AT END imperative-statement-2 ] [ END-READ ]
```

An example is given below. ACCESS IS SEQUENTIAL

```
READ PERSONNEL-FILE RECORD
     INTO WS-PERS-RECORD
     AT END PERFORM INPUT-COMPLETED.
```

. If the file access mode is DYNAMIC, the sequential READ requires the use of the NEXT phrase.

```
READ PERSONNEL-FILE NEXT RECORD
     INTO WS-PERS-RECORD
     AT END PERFORM INPUT-COMPLETED.
```

Random-access input requires a different syntax

ACCESS IS RANDOM

```
READ file-name RECORD [ INTO identifier ]
    [ KEY IS data-name ]
    [ INVALID KEY imperative-statement-1 ]
    [ NOT INVALID KEY imperative-statement-2 ]
    [ END-READ ]
```

```
 ...................
 ENVIRONMENT DIVISION.
 ...................
 SELECT PERSONNEL-FILE
     ASSIGN                    TO INFILE
     ORGANIZATION              IS INDEXED
     ACCESS                    IS RANDOM
     RECORD KEY                IS EMPLOYEE-NUMBER
     ALTERNATE RECORD KEY IS EMPLOYEE-AGE WITH DUPLICATES
     ALTERNATE RECORD KEY IS DEPARTMENT-CODE WITH DUPLIC
 ...................
 PROCEDURE DIVISION.
 ...................
***** RANDOM RETRIEVAL BY PRIMARY KEY, "EMPLOYEE-NUMBER"

       MOVE "1200" TO EMPLOYEE-NUMBER.

       READ PERSONNEL-FILE
          KEY IS EMPLOYEE-NUMBER
          INVALID KEY PERFORM ERROR-IN-INPUT.
 ...................
***** RANDOM RETRIEVAL BY ALTERNATE KEY, "EMPLOYEE-AGE"

       MOVE 29 TO EMPLOYEE-AGE.

       READ PERSONNEL-FILE
          KEY IS EMPLOYEE-AGE
          INVALID KEY PERFORM ERROR-IN-INPUT.
 ...................
```

WRITE Statement. There is only one syntax for the WRITE statement for indexed sequential files. It resembles the WRITE statement for sequential files except for the optional INVALID KEY clause.

COBOL 85

```
WRITE record-name [ FROM identifier ]
    [ INVALID KEY imperative-statement-1 ]
    [ NOT INVALID KEY imperative-statement-2 ] [ END-WRITE ]
```

An example is provided below.

```
WRITE PERSONNEL-RECORD
   FROM WS-PERS-RECORD
   INVALID KEY PERFORM ERROR-IN-OUTPUT.
```

Under the following circumstances, an output operation will fail and an invalid key condition will be raised:

1. Key order violation during loading
2. Record key duplication during insertion
3. Alternate key duplication
4. Attempting to write into a full file

REWRITE Statement. The purpose of the REWRITE statement for indexed files is to update a record. Syntactically, it resembles the WRITE statement.

```
REWRITE record-name [ FROM identifier ]
        [ INVALID KEY imperative-statement-1 ]
        [ NOT INVALID KEY imperative-statement-2 ]
        [ END-REWRITE ]
```

```
........................
MOVE "1200" TO EMPLOYEE-NUMBER.

READ PERSONNEL-FILE
   INVALID KEY PERFORM ERROR-IN-INPUT.

MOVE "SHEILA GORDON" TO EMPLOYEE-NAME.
ADD 1 TO EMPLOYEE-AGE.
........................
REWRITE PERSONNEL-RECORD
   INVALID KEY PERFORM ERROR-IN-REWRITE.
```

If the file access mode is RANDOM, a random READ should be used to access a record. An invalid key condition will be raised with REWRITE under two circumstances.

1. The primary key value of the record that has been read most recently is changed prior to rewriting.
2. An alternate key value of the record has been changed such that rewriting violates its uniqueness. This happens if the alternate key has been defined without a WITH DUPLICATES clause.

DELETE Statement. This statement is used to access a record either sequentially or directly and delete it from the file. Syntactically, the DELETE statement is defined as follows.

```
DELETE file-name RECORD
    [ INVALID KEY imperative-statement-1 ]
    [ NOT INVALID KEY imperative-statement-2 ] [ END-DELETE ]
```

An example is provided below.

```
........................
MOVE "1200" TO EMPLOYEE-NUMBER.

DELETE PERSONNEL-FILE
   INVALID KEY PERFORM ERROR-IN-DELETE.
```

```
                          ⎧ EQUAL TO                        ⎫
                          ⎪ =                               ⎪
                          ⎪ GREATER THAN                    ⎪
                          ⎪ >                               ⎪
[START file-name  KEY IS  ⎨                                 ⎬ data-name ]
                          ⎪ NOT LESS THAN                   ⎪
                          ⎪ NOT <                           ⎪
                          ⎪ GREATER THAN OR EQUAL TO        ⎪
                          ⎩ >=                              ⎭
        [ INVALID KEY  imperative-statement-1 ]
        [ NOT INVALID KEY  imperative-statement-2 ]  [ END-START ]
```

```
.........................
MOVE "1200" TO EMPLOYEE-NUMBER.

START PERSONNEL-FILE
   KEY IS GREATER THAN EMPLOYEE-NUMBER
   INVALID KEY PERFORM ERROR-IN-START.

READ PERSONNEL-FILE
    AT END MOVE "YES" TO END-OF-FILE-FLAG.
.........................
```

The START statement above locates the first record whose primary key value is larger than "1200", and the READ statement accesses that record.

Here is another example.

```
.........................
MOVE "1200" TO EMPLOYEE-NUMBER.

START PERSONNEL-FILE
   INVALID KEY PERFORM ERROR-IN-START.

READ PERSONNEL-FILE
    AT END MOVE "YES" TO END-OF-FILE-FLAG.
.........................
```

In this case, the START predicate is implicit and is equivalent to KEY IS EQUAL TO EMPLOYEE-NUMBER. Therefore, the READ statement will access the record whose primary key value is equal to "1200".

An invalid key condition will be raised if the START predicate cannot be satisfied within the scope of the key space for the file. For example, if the START predicate contains IS EQUAL TO and there is no record in the file whose key value matches that of the current value stored in the data name in the predicate, the invalid key condition is true. As another example, if the START predicate is KEY IS GREATER THAN EMPLOYEE-NUMBER, the current value of EMPLOYEE-NUMBER is 1200, and the largest key value in the file is 1100, again an invalid key condition will be raised.